**Weather Data Analysis using Methods**

**Description**:

Your task is to analyze the weather data between January 2020 to May 2020 from the Buttonville weather station located between Markham and Richmond Hill. The goal is to provide an analysis of the weather patterns in the year of 2020 and to identify any potential extreme weather events during this time. You are provided data from the weather station in this timeframe as well as a java program containing the main created by another programmer.

**Program Requirements:**

Each feature of the program can be broken down into parts to indicate the scale and subsequent level. Your program should be able to:

1. Open the data and store it in an appropriate data structure
2. Calculate and output the average temperature for…
   1. Level 1: Daily data for the month of May
   2. Level 2: Daily data for the month of May as well as an analysis of the morning, afternoon, and evening for each day
   3. Level 3: Daily and weekly analysis for the month of May
   4. Level 4: Daily, weekly, monthly, and annually analysis for the year of 2020 based on the data provided
3. Calculate/Output the maximum and minimum temperatures for the time period of the level you’ve selected and completed
4. **Level 4 Additional Expectation**: Identify extreme weather alerts based on parameters such as high/low precipitation, wind speed, and/or temperature
5. Output only the results of you analysis in…
   1. Level 1 – 3: The terminal/shell of the java IDE in an organized and formatted fashion
   2. Level 4: In a formatted file named “results.csv” (Consider opening the data provided in notepad to see what the formatting of a .csv file looks like)
6. All methods created must contain fully documented Javadoc strings <https://en.wikipedia.org/wiki/Javadoc#Structure_of_a_Javadoc_comment>

\*Note that there will be **no** user input elements for this program. This is a pure processing exercise and should process the data once the user clicks “run” in the IDE

**Submission Requirements**:

It is critical that the main method of the program is **NOT** changed in this program unless you are attempting the level 4 of this program or the main specifies to change it.

If you do attempt/complete the level 4 of this project, you **must** submit a working level 3 along with your level 4 submission. Failure to submit the level 3 work will result in penalties subject to your instructor.

**General Criteria**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Criteria | Level 1 | Level 2 | Level 3 | Level 4 |
| Data Processing | Program capable of completing a daily analysis | Program capable of completing analysis for the morning, afternoon, and evening of each day and a daily analysis  Program outputs the result | Program capable of completing the analysis for the first week of May  Program outputs results in the terminal in a formatted manner | Program capable of completing the analysis for the whole year of 2020  Processed data is summarized results in a formatted table and outputted in a separate .csv file |
| Use of Methods | Methods are created to accomplish the task | Methods are created to be effective and efficient at solving the problem | Appropriate number of additional methods used to streamline existing methods in program | All of level 3 criteria plus  Minimal changes to the main to achieve level 4 criteria in data processing |
| Modular Programming | Significant duplication of algorithms and/or data structures. | Some duplication of algorithms and/or data structures. | Minimal duplication of algorithms and/or data structures. | **No unnecessary duplication** of **algorithms** and/or data structures. |
| Internal Documentation and Doc Strings | Minimal internal documentation is provided.  Minimal or no Javadoc/docstrings. | Minimal internal documentation is provided.  Javadoc/docstrings is incomplete. | Internal documentation is provided, but incomplete or confusing.  Javadoc/docstrings for most methods and classes. | Thorough **internal documentation** is provided and it **clearly explains** the design.  **Javadoc/docstrings** for each class and method |

**Extra Resources**

Data source:

bit.ly/buttonvilleweather

Some Useful Methods:

**public** String[] split(String regex, **int** limit) {

**Parameters:**

regex - a delimiting regular expression

Limit - the result threshold

**Returns:**

An array of strings computed by splitting the given string.

static int parse(String s) {

**Parameters:**

s – This is a string representation of decimal

**Returns:**

An integer (decimal only)